Kernelized SVM for classification

Kernel trick: replacing the linear transformation that we usually have in linear models with a Kernel - any function that measures the distance between two vectors.

Linear models defined in terms of the Kernel function:

$$y(x;\alpha) = \sum_{n=1}^{N} \alpha_n k(x_n, x)$$

$$y(x;\alpha) = \sum_{n=1}^{N} \alpha_n x_n^T x$$

linear model with linear Kernel

Solution

$$\alpha = (K + \lambda I_N)^{-1} t$$

Gram matrix

$$K = \begin{bmatrix} k(x_1,x_1) & \cdots & k(x_1,x_N) \\ \vdots & \ddots & \vdots \\ k(x_N,x_1) & \cdots & k(x_N,x_N) \end{bmatrix}$$

The evaluation of α would be almost infeasible but an important property of SVMs is that the computation of the model parameters corresponds to a convex optimization problem and so any local solution is also a global optimum.

Previous formulation of the SVMs:

$$y(x;w) = w^T x + w_0$$

$$\hat{w} = \sum_{n=1}^{N} a_n^* t_n x_n$$

$$y(x) = w_0 + \sum_{x_n \in SV} a_n^* t_n x^T x_n$$

Hyperplanes expressed with support vectors $\quad y(x) = w_0 + \sum_{x_j \in SV} \underbrace{\boxed{a_j^* t_j}}_{\alpha_j} x^T x_j = 0$

Samples on one side are classified one way, samples on the other side are classified the other way

$$y(x';\alpha) = \text{sign}\left(w_0^* + \sum_{x_k \in SV} \underbrace{\boxed{a_k^* t_k}}_{\alpha_k} x'^T x_k\right)$$

linear Kernel

Solution of the problem:

$$w^* = \sum_{n=1}^{N} \underbrace{\boxed{a_n^* t_n}}_{\alpha_n} x_n$$

Now with Kernelization:

$$x_i^T x_j \longleftarrow k(x_i, x_j)$$

Classification:

$$y(x';\alpha) = \text{sign}\left(w_0 + \sum_{n=1}^{N} \alpha_n k(x_n, x)\right)$$

generic Kernel